Machine and Deep Learning Data Challenge: Sub-event Detection in Twitter Streams

Team: Sigmoid Sigmas

Tim Luka Horstmann
Institut Polytechnique de Paris
tim.horstmann@ip-paris.fr

Mathieu Antonopoulos
Institut Polytechnique de Paris
Antonopoulos@ip-paris.fr

Baptiste Geisenberger
Institut Polytechnique de Paris
geisenberger@ip-paris.fr

Abstract

This report presents a comprehensive approach to sub-event detection in Twitter streams, undertaken as part of CSC_51054_EP's Machine and Deep Learning Data Challenge. The challenge focuses on analyzing tweets posted during football matches from the 2010 and 2014 FIFA World Cups to predict the presence of specific sub-events in one-minute time periods. Leveraging the rich dynamics of crowd interactions on social media, we employ a multi-faceted methodology that explores traditional machine learning (ML), graph-based models, and large language models (LLMs) to classify sub-events. We present robust preprocessing and feature engineering approaches, as well as different model development phases, leveraging techniques such as boosting, graph representation, pre-computed embeddings, and fine-tuning of LLMs. We achieve the best results with our *EnhancedPeriodClassifier* model, which integrates pre-computed embeddings with supplementary temporal and contextual features for improved computational efficiency and accuracy in sub-event detection. Comprehensive evaluations of all our approaches, including feature importance analysis, ablation studies, and hyperparameter optimization, highlight the significance of different domain-specific features. Our study demonstrates not only the effectiveness of a variety of ML techniques on the challenging task of understanding high-volumes of noisy social media data, but also the challenges these models face in such contexts.

1 Data Preprocessing and Feature Selection/Extraction

1.1 Data Exploration and Preprocessing

Following an extensive data exploration phase, where we analyzed the tweet data made available as part of this data challenge in regards to temporal distribution of tweets, class distribution, duplicates, and other relevant features, we built upon the preprocessing methodology introduced in related research by Meladianos et al. [17, 16], merging key ideas to suit our three approaches to sub-event detection (Section 2). First, we clean the dataset, starting with tokenization, where each tweet is split into individual words (tokens). To ensure uniformity and avoid case sensitivity inconsistencies, all tokens are converted to lowercase. Research has shown that retweets introduce noise [17]. Hence, we remove retweets and duplicates globally. Additionally, tweets that contain mentions were excluded, as these are often unrelated to the event of interest [17]. To further refine the dataset, stopwords are removed, alongside any special characters and URLs, which are common sources of noise [16]. This methodology was followed for our graph-based and traditional machine learning (ML) approaches, but adapted for our large language models (LLMs) approach to balance noise removal with context preservation (e.g. keeping stopwords).

1.2 Feature Engineering

Feature engineering is a pivotal step, particularly for the methods outlined in Sections 2.1 and 2.3. Indeed, this step transforms raw data into structured, meaningful inputs that the models can learn from, enhancing performance. Inexpensive to implement, we extracted a variety of features. Table 1 provides an overview of the most notable features we worked with — either those that contributed significantly to accuracy or those that illustrate relevant intuitions.

Type	ID	Feature	Rationale	Implementation	Related Work
Language Encoding	L1	Word embeddings	Captures contextualised semantic meaning from tweets.	TF-IDF, Encoder LLMs	[3, 5, 20, 23]
Temporal Features	T1	Tweet Volume	Hypothesized higher volume during notable events.	Total number of tweets per time period.	[2]
	T2	Sudden Increase	Highlights abrupt spikes in activity, potentially indicating a sub-event.	Tweet volume variation compared to the previous period.	[14]
Contextual Features	C1	Period Number	Encodes temporal markers correlated with event timings.	ID of the corresponding period.	[1]
	C2	Tweet Length	Hypothesized to correlate with emotional responses.	Maximum and average tweet lengths per period.	[19]
	C3	Keyword Count	Indicates semantic relevance.	Keyword Count per sub-event	[15, 2]
	C4	Sentiment Score	Indicates semantic relevance.	NLTK Sentiment Intensity Analyzer [10]	[19, 11]

Table 1: Overview of the engineered features used as basis for our sub-event detection approaches.

We monitored the impact of these features using ablation studies and model-integrated scores. Further details are provided in the respective sections of this report.

2 Model Choice, Tuning and Comparison

2.1 Sub-event Detection in Twitter Streams via Traditional Machine Learning Models

In our initial approach, we created a feature matrix and employed numerous traditional ML models, summarized in Table 2. This well-established method included phases of feature engineering, model selection, and hyperparameter optimization (HPO), as outlined in Figure 1 [21].

Our feature engineering (see Section 1.2) led us to incorporate all features but sentiment analysis (C5) of Table 1 and more. We employed Twitter-RoBERTa [13], creating rich embeddings, and averaging them per period to aggregate semantic meaning. Our extracted features were relevant in most cases, particularly keyword count (C4) - a feature we designed to help the model identify keywords relevant to specific sub-event types.

We applied this approach using various basic traditional models from the scikit-learn library[1], and boosting techniques that offer efficient implementations/libraries. We scaled the features through standardization or min-max scaling, yielding better results in logistic regression and K-nn, while reducing computation time in general.

Assessing feature importance is crucial for identifying relevant information encapsulated in the data and removing noise, thus improving performance [8]. We used model-integrated scores to evaluate individual feature importance and ablated irrelevant features to assess potential accuracy gains.

[1] `https://scikit-learn.org/stable/index.html`

Following best practices in research, we conducted an extensive HPO through grid search for each of our models [24].

Model	Key Parameters	Accuracy (%)
Logistic Regression	Regularization strength (C): 0.1, Penalty: L2	
K-Nearest Neighbors (KNN)	(k): 7, Distance: Minkowski (p = 4)	0.68 validation data (VD)
Random Forest	Number of trees: 100, Maximum depth: 5	
LightGBM	Maximum depth: 10, Fraction of features selected by tree: 0.6	0.68 (VD)
Adaboost	Number of trees: 1000, Learning rate: 0.01	
Catboost	Number of iterations: 100, Depth of trees: 5	
SVM (Linear Kernel)	Regularization strength: 10, DimRed: Nyström	0.51
SVM (RBF Kernel)	Regularization strength: 10, DimRed: Nyström	0.52

Table 2: Accuracy of traditional models with key parameters (on test set from train data)

2.2 Sub-event Detection in Twitter Streams via Graph Models

To leverage the structural relationships between tweets, we represented the data as graphs to detect sub-events. After preprocessing (see Section 1), each tweet was transformed into a fully connected graph where unique words served as vertices and the weights of edges reflected their co-occurrence frequency. These weights were normalized by the tweet's length to account for varying tweet lengths. Subsequently, we aggregated all tweet-level graphs for a given time period to construct an adjacency matrix, which formed the basis for sub-event detection.

2.2.1 Deviation analysis

The first detection method focused on identifying deviations in graph structures. Specifically, we compared the adjacency matrix of the current time period to the aggregated matrices of the preceding "P" periods (e.g., P = 12). Using least-squares optimization, the edge weights of the current period's graph were fitted to those of prior periods. A significant deviation from the aggregated structure, as determined by a predefined threshold, indicated the occurrence of a sub-event. This approach emphasized detecting abrupt structural shifts within the temporal graph sequence, consistent with methodologies used in prior research [3, 17, 16]. This method was adapted from the approach described in Meladionis et al. [16].

2.2.2 Graph evolution analysis

The second method analyzed changes in adjacency matrices between consecutive periods. By calculating the Frobenius norm of the difference between matrices, we measured the degree of structural evolution. If the computed norm exceeds a threshold, a change in event type is detected.

Method	Deviation Analysis	Graph Evolution Analysis
Scope	Compares current period to P previous periods	Compares consecutive periods
Optimization Method	Least-squares optimization	Frobenius norm of adjacency matrix
Focus	Sudden changes in specific weights	Average changes in weights

Table 3: Comparison of deviation and graph evolution analysis

Both methods demonstrated potential for sub-event detection. However, neither of them proved to be highly effective on the test set. The deviation analysis achieved an accuracy of 0.62, which is similar to the frequency classifier. This relatively low accuracy can be attributed to the lack of dynamic thresholding, which would require a computationally expensive training phase. This limitation made it infeasible to implement within the available time frame given our implementation. Future iterations could implement dynamic thresholding as explored by Meladionis et al [16].

2.3 Sub-event Detection in Twitter Streams via Large Language Models

As part of our third approach to sub-event detection, we experimented with fine-tuning pre-trained LLMs for period classification — an approach shown to generally yield strong results [25] — as well as using pre-computed embeddings for downstream classification to reduce computational costs. Initially, fine-tuning at the tweet level with majority voting was also tested but found less effective. Experiments were conducted on NVIDIA RTX A4000 GPUs and meticulously monitored via Weights & Biases[2], where we tracked over 150 hours of model training, corresponding logs, and files to ensure reproducibility of each of our experiments.

[2] `https://wandb.ai/site/`

2.3.1 The *PeriodClassifier*: a fine-tuning approach

To classify time periods comprising multiple tweets, we implemented two fine-tuning approaches: tokenizing tweets individually and aggregating mean embeddings with attention, or processing concatenated tweets as sequences. Given the domain-specific nature of the data challenge, we chose BERTweet[18] for the first approach and Longformer [4] for the second to handle the longer context-window required by concatenation. Although promising, fine-tuning proved computationally expensive and less effective than the embeddings-based approach described in Section 2.3.2.

2.3.2 The *EnhancedPeriodClassifier*: a pre-computed embeddings approach

To improve efficiency, we designed a custom model using pre-computed BERTweet embeddings. First, we tokenize the preprocessed tweets (see Section 1) using BERTweet's tokenizer, leveraging its additional normalization qualities, and pass up to `MAX_TWEETS` (`=1500`) tokenized tweets per period through BERTweet to obtain accurate tweet representations in latent space. We represent individual tweets by their mean-pooled embeddings, which may better represent short and noisy data compared to using [CLS] embeddings, as done in Devlin et al. [6], for example. These embeddings are aggregated via Multi-Head Attention[22] and learnable pooling in our custom *EnhancedPeriodClassifier*. We enrich the resulting period embeddings with additional normalized features (see Section 1.2) before classification via a three-layer neural network. Figure 3 outlines this approach.

2.3.3 Ablation Study

To identify the best model and mitigate overfitting, we conducted a comprehensive optimization study (see Figure 4). We first compared the different architectures described in this section pre-HPO. The pre-computed embeddings approach based on BERTweet emerged as the best model while fine-tuning Twitter-RoBERTa exhibited strong promise. However, since its architecture and tokenization differs to the other models, exploring this approach is kept for future work. Opting for the *EnhancedPeriodClassifier* model, we explored the effects of adding different additional features (T1, C1, C2, C4) to the aggregated period embeddings through an ablation study (see Table 1). Allowing the model to specifically consider T1, for instance, improved performance while adding C4 did not — likely due to the model's implicit sentiment understanding or limitations of the sentiment analyzer we employed.

2.3.4 Hyperparameter Optimization Study

We employed the Optuna[3] framework for an efficient and systematic exploration of our final model's hyperparameter space. Concretely, we evaluated the learning rate, number of attention heads, dimension of the first fully connected layer in the classification head (`fc1_dim`), setting the dimension of `fc2_dim` to be half of this value, dropout probability, and weight decay, with the latter two specifically introduced to prevent overfitting. Due to the high computational costs of large-scale HPO, we trained 30 different *EnhancedPeriodClassifier* models over the course of two days, exploring a variety of hyperparameters, with `fc1_dim` being identified as the most important hyperparameter in our experiments (Figure 5). This highlights the critical role of the classification head in determining the model's accuracy, as it directly influences the model's capacity to process and interpret the rich contextualized embeddings generated in the earlier steps.

3 Final Model Results and Evaluation

Comparing the three different approaches to sub-event detection outlined in Section 2, we found the graph-based model (Section 2.2) to be too computational expensive for this data challenge, while the traditional ML methods (Section 2.1) delivered good results, achieving an accuracy of ca. 72% on the (Kaggle) test set. The *EnhancedPeriodClassifierModel* (Section 2.3.2) emerged as the most successful approach with an accuracy of ca. 75%. However, it should be noted that this approach is generally more complex and computational more expensive than the traditional ML methods.

Building on the generally strong performance of the LLM approach, we briefly explored a further modified version of this model[4], adding additional components for improved robustness, such as residual connections, additional pooling, deeper neural architectures, and switching to a GELU activation function, due its often superior performance compared to ReLu [12]. This is also the model achieving our team's final Kaggle result, with an accuracy of 77% on the test set. This model could be further explored and optimized in the future.

4 Conclusion and Future Work

This report presented an end-to-end method for sub-event detection in Twitter streams using data from the 2010 and 2014 FIFA World Cups. By employing traditional, graph, and large language models, we successfully identified significant events, achieving a final accuracy of ca. 77% on this data challenge's test set. The *EnhancedPeriodClassifier*, a LLM approach based on enriching pre-computed embeddings with additional hand-crafted features, proved the most effective. Feature engineering, ablation studies, and hyperparameter optimization highlighted the crucial role of both the preparatory and post-model processes in enhancing model performance. Future work will explore additional features, refine the presented ML methods through more extensive HPO, and investigate advanced model architectures, building on the strengths of our three approaches and paving the way for further research into ML applications for social media data.

[3] `https://optuna.org/`

[4] The model's training performance is available at `https://api.wandb.ai/links/ml-data-challenge-24/ndamu0k0`.

References

[1] D. Abhik and D. Toshniwal. Sub-event detection during natural hazards using features of social media data. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13 Companion, pages 783–788, New York, NY, USA, May 2013. Association for Computing Machinery. ISBN 978-1-4503-2038-2. doi: 10.1145/2487788.2488046. URL https://dl.acm.org/doi/10.1145/2487788.2488046.

[2] H. Becker, M. Naaman, and L. Gravano. Beyond Trending Topics: Real-World Event Identification on Twitter. *Proceedings of the International AAAI Conference on Web and Social Media*, 5(1):438–441, 2011. ISSN 2334-0770. doi: 10.1609/icwsm.v5i1.14146. URL https://ojs.aaai.org/index.php/ICWSM/article/view/14146. Number: 1.

[3] G. Bekoulis, J. Deleu, T. Demeester, and C. Develder. Sub-event detection from twitter streams as a sequence labeling problem. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 745–750, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1081. URL https://aclanthology.org/N19-1081.

[4] I. Beltagy, M. E. Peters, and A. Cohan. Longformer: The Long-Document Transformer, Dec. 2020. URL http://arxiv.org/abs/2004.05150. arXiv:2004.05150 [cs].

[5] G. Chen, N. Xu, and W. Mao. An Encoder-Memory-Decoder Framework for Sub-Event Detection in Social Media. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 1575–1578, New York, NY, USA, Oct. 2018. Association for Computing Machinery. ISBN 978-1-4503-6014-2. doi: 10.1145/3269206.3269256. URL https://doi.org/10.1145/3269206.3269256.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. URL http://arxiv.org/abs/1810.04805. arXiv:1810.04805 [cs].

[7] G. Glavaš and S. Somasundaran. Two-Level Transformer and Auxiliary Coherence Modeling for Improved Text Segmentation, Jan. 2020. URL http://arxiv.org/abs/2001.00891. arXiv:2001.00891 [cs].

[8] I. Guyon and A. Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research 3*, 2003.

[9] H. Hettiarachchi, M. Adedoyin-Olowe, J. Bhogal, and M. M. Gaber. DAAI at CASE 2021 Task 1: Transformer-based Multilingual Socio-political and Crisis Event Detection. In A. Hürriyetoğlu, editor, *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*, pages 120–130, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.case-1.16. URL https://aclanthology.org/2021.case-1.16.

[10] C. Hutto and E. Gilbert. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1):216–225, May 2014. ISSN 2334-0770. doi: 10.1609/icwsm.v8i1.14550. URL https://ojs.aaai.org/index.php/ICWSM/article/view/14550. Number: 1.

[11] T. Kolajo, O. Daramola, and A. A. Adebiyi. Real-time event detection in social media streams through semantic analysis of noisy terms. *Journal of Big Data*, 9(1):90, July 2022. ISSN 2196-1115. doi: 10.1186/s40537-022-00642-y. URL https://doi.org/10.1186/s40537-022-00642-y.

[12] M. Lee. GELU Activation Function in Deep Learning: A Comprehensive Mathematical Analysis and Performance, Aug. 2023. URL http://arxiv.org/abs/2305.12073. arXiv:2305.12073 [cs].

[13] D. Loureiro, K. Rezaee, T. Riahi, F. Barbieri, L. Neves, L. E. Anke, and J. Camacho-Collados. Tweet Insights: A Visualization Platform to Extract Temporal Insights from Twitter, Aug. 2023. URL http://arxiv.org/abs/2308.02142. arXiv:2308.02142 [cs].

[14] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller. Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 227–236, New York, NY, USA, May 2011. Association for Computing Machinery. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1978975. URL https://doi.org/10.1145/1978942.1978975.

[15] M. Mathioudakis and N. Koudas. TwitterMonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158, Indianapolis Indiana USA, June 2010. ACM. ISBN 978-1-4503-0032-2. doi: 10.1145/1807167.1807306. URL https://dl.acm.org/doi/10.1145/1807167.1807306.

[16] P. Meladianos, C. Xypolopoulos, G. Nikolentzos, and M. Vazirgiannis. An Optimization Approach for Sub-event Detection and Summarization in Twitter. In G. Pasi, B. Piwowarski, L. Azzopardi, and A. Hanbury, editors, *Advances in Information Retrieval*, volume 10772, pages 481–493. Springer International Publishing, Cham, 2018. ISBN 978-3-319-76940-0 978-3-319-76941-7. doi: 10.1007/978-3-319-76941-7_36. URL https://link.springer.com/10.1007/978-3-319-76941-7_36. Series Title: Lecture Notes in Computer Science.

[17] P. Meladianos, G. Nikolentzos, F. Rousseau, Y. Stavrakas, and M. Vazirgiannis. Degeneracy-Based Real-Time Sub-Event Detection in Twitter Stream. *Proceedings of the International AAAI Conference on Web and Social Media*, 9(1):248–257, Aug. 2021. ISSN 2334-0770, 2162-3449. doi: 10.1609/icwsm.v9i1.14597. URL https://ojs.aaai.org/index.php/ICWSM/article/view/14597.

[18] D. Q. Nguyen, T. Vu, and A. Tuan Nguyen. BERTweet: A pre-trained language model for English Tweets. In Q. Liu and D. Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online, Oct. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.2. URL https://aclanthology.org/2020.emnlp-demos.2.

[19] T. Nguyen. A Comprehensive Low and High-level Feature Analysis for Early Rumor Detection on Twitter, Apr. 2024. URL http://arxiv.org/abs/1711.00726. arXiv:1711.00726 [cs] version: 3.

[20] C. Shen, F. Liu, F. Weng, and T. Li. A Participant-based Approach for Event Summarization Using Twitter Streams. In L. Vanderwende, H. Daumé III, and K. Kirchhoff, editors, *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1162, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL https://aclanthology.org/N13-1135.

[21] J. P. Singh, Y. K. Dwivedi, N. P. Rana, A. Kumar, and K. K. Kapoor. Event classification and location prediction from tweets during disasters. *Annals of Operations Research*, 283(1):737–757, Dec. 2019. ISSN 1572-9338. doi: 10.1007/s10479-017-2522-3. URL https://doi.org/10.1007/s10479-017-2522-3.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need, June 2017. URL http://arxiv.org/abs/1706.03762. arXiv:1706.03762 [cs] version: 1.

[23] Z. Wang and Y. Zhang. A neural model for joint event detection and summarization. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pages 4158–4164, Melbourne, Australia, Aug. 2017. AAAI Press. ISBN 978-0-9992411-0-3.

[24] L. Yang and A. Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415: 295–316, Nov. 2020. ISSN 0925-2312. doi: 10.1016/j.neucom.2020.07.061. URL https://www.sciencedirect.com/science/article/pii/S0925231220311693.

[25] Y. Zhou and V. Srikumar. A Closer Look at How Fine-tuning Changes BERT, Mar. 2022. URL http://arxiv.org/abs/2106.14282. arXiv:2106.14282 [cs].

Appendices

A Architecture of the Sub-event Detection Approaches

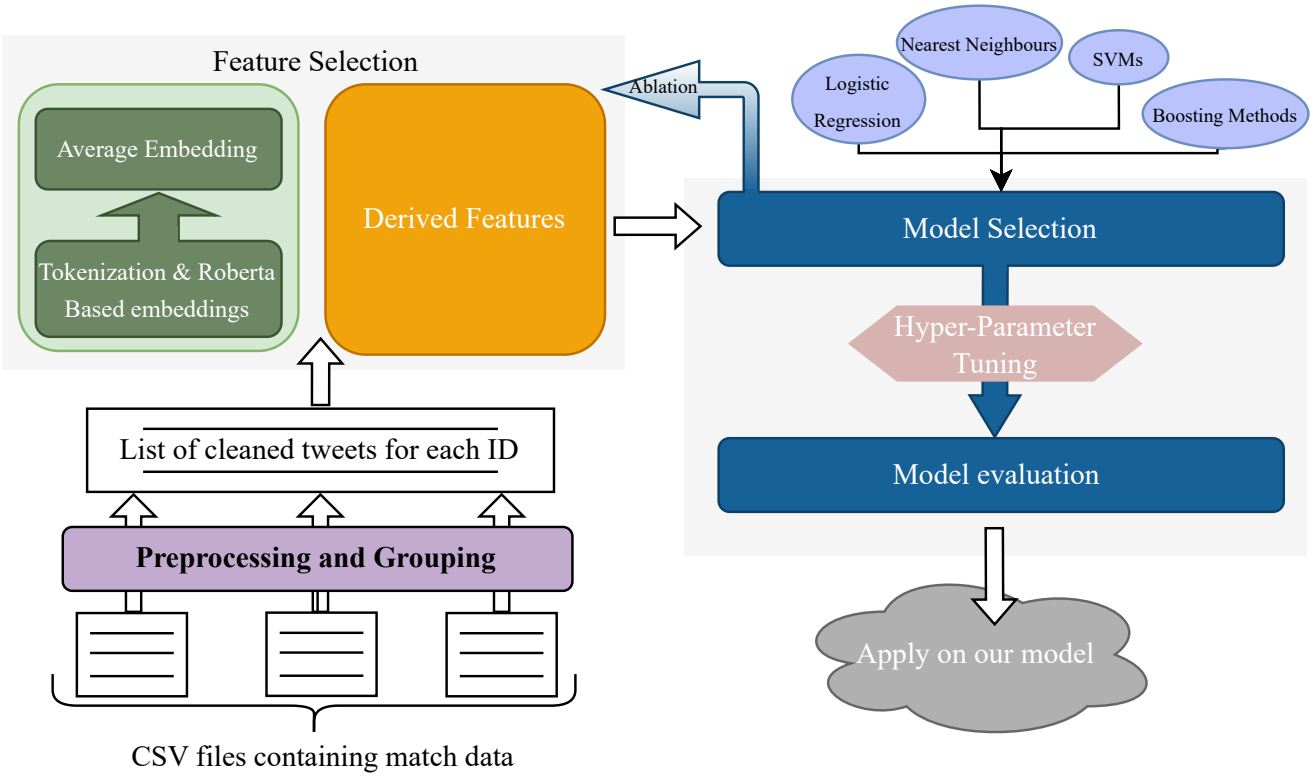


Figure 1: Architecture of the traditional model approach.

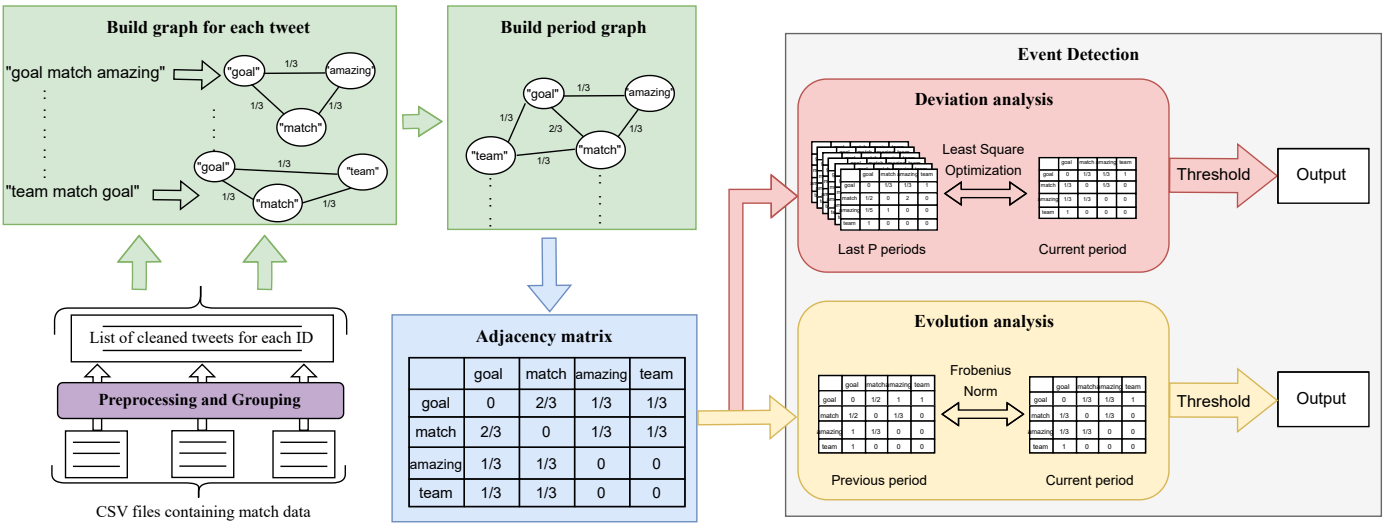


Figure 2: Architecture of the graph-based approach.

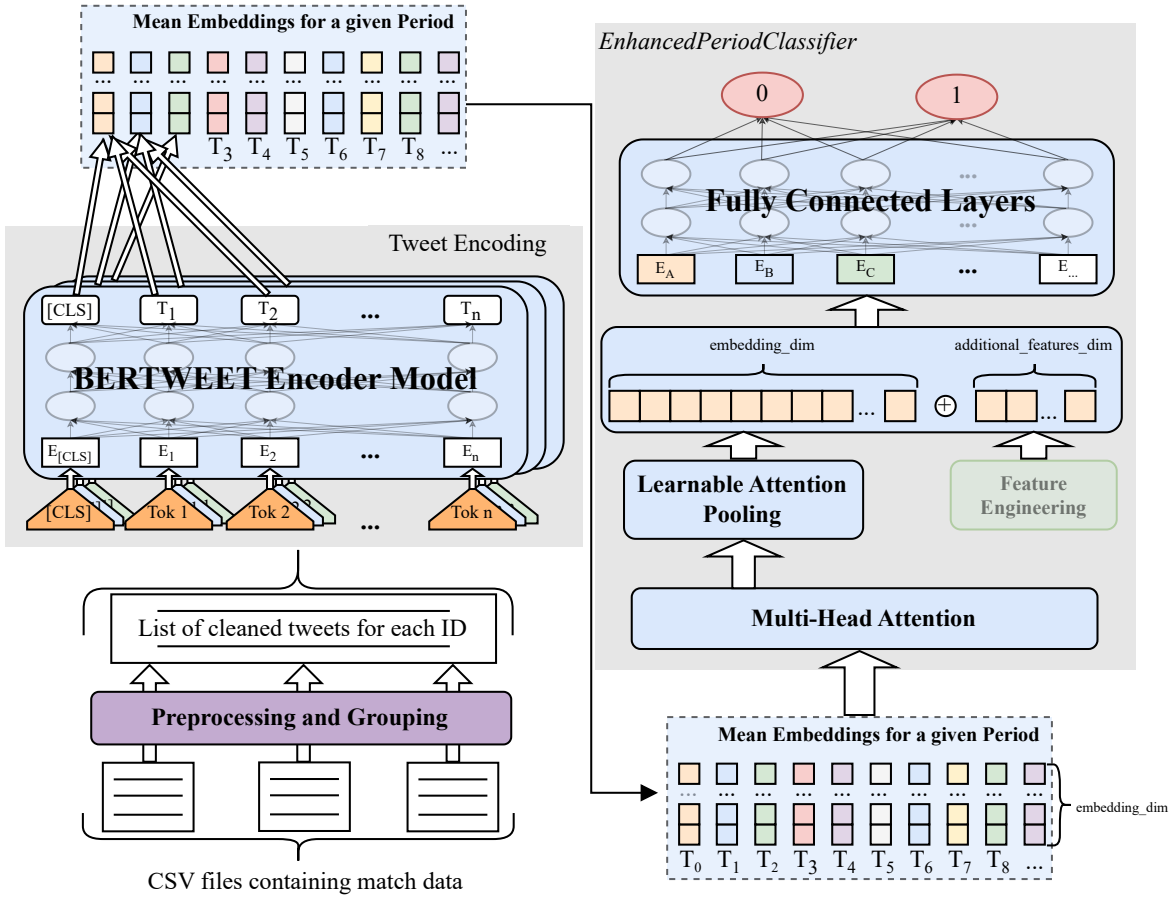
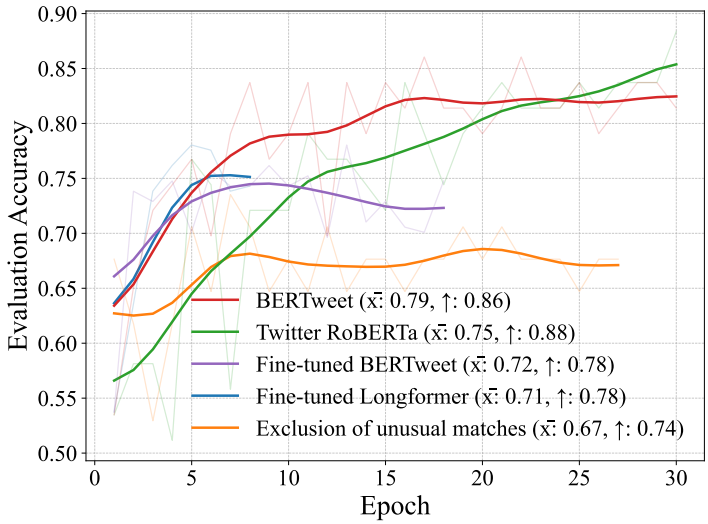
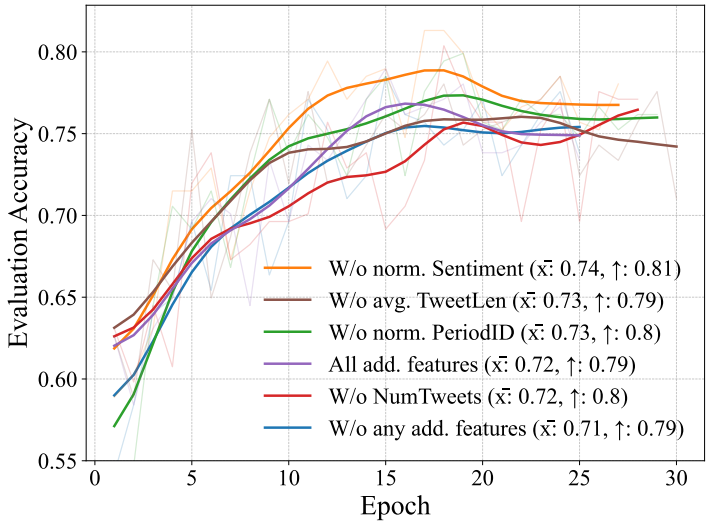


Figure 3: Architecture of the *EnhancedPeriodClassifier* model. Graphic based on [9, 7].

B Model Selection and Ablation Study Results of the LLM-based Approaches



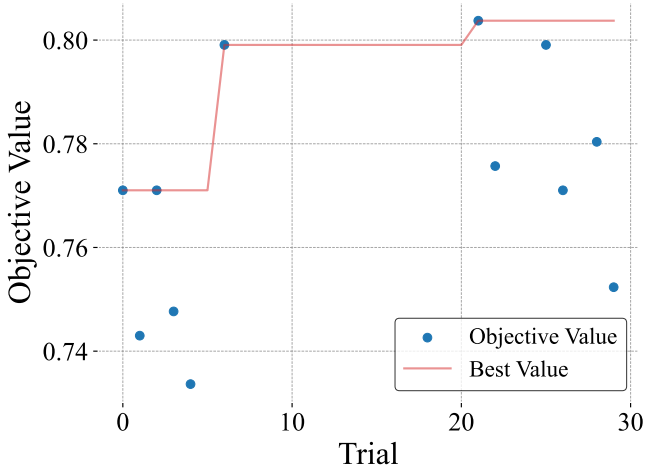
(a) Comparison of different LLM architectures. Best results were obtained by employing the frozen BERTweet model. Twitter RoBERTa shows strong promise.



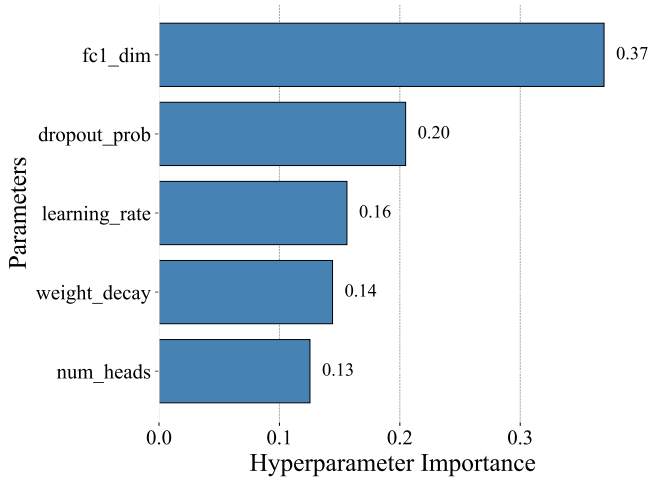
(b) Ablation study of additional features to enrich pre-computed BERTweet embeddings. Best results were obtained without the normalized sentiment feature.

Figure 4: Model selection and ablation study results: evaluation accuracy per epoch with 20% and 10% held-out validation data, respectively, for original and Gaussian-smoothed results.

C Results of EnhancedPeriodClassifierModel HPO



(a) Optimization history. The best trial was Trial 21 with a validation accuracy of around 80%.



(b) Hyperparameter importance. The dimension of the first fully connected layer, which also defines the dimensions of the second, is considered the most important hyperparameter.

Figure 5: Results of the HPO using Optunaf. We identified the following best parameters for our model: `lr=0.000207, dropout=0.1, num_heads=12, weight_decay=0.0023, fc1_dim=2048`

Trial	Accuracy	Dropout	FC1 Dim	Learning Rate	Num Heads	Weight Decay
0	0.771	0.48	1024	5.6e-5	4	4.2e-6
1	0.743	0.49	1024	1.1e-5	4	5.4e-6
2	0.771	0.16	2048	1.7e-4	12	1.4e-3
3	0.748	0.17	2048	1.6e-4	12	1.7e-3
4	0.734	0.3	2048	1.8e-5	8	4.5e-4
5	0.589	0.41	4096	8.7e-4	4	4.9e-3
6	0.799	0.21	4096	6.0e-5	4	1.5e-4
7	0.617	0.18	512	3.5e-4	8	8.2e-4
8	0.654	0.35	2048	5.3e-4	4	2.0e-5
9	0.407	0.39	512	1.7e-5	12	9.4e-5
10	0.659	0.25	4096	5.8e-5	4	6.1e-5
11	0.584	0.49	1024	4.9e-5	4	1.4e-6
12	0.72	0.25	1024	4.9e-5	4	1.5e-5
13	0.621	0.24	4096	9.0e-5	4	1.1e-6
14	0.57	0.11	1024	3.0e-5	4	2.3e-4
15	0.612	0.31	4096	1.5e-4	8	2.9e-5
16	0.715	0.41	4096	7.5e-5	4	3.0e-6
17	0.575	0.45	1024	3.0e-5	4	2.1e-4
18	0.636	0.3	512	3.3e-4	12	5.6e-6
19	0.603	0.21	4096	3.3e-5	4	3.6e-5
20	0.612	0.34	1024	1.3e-4	8	8.1e-3
21	0.804	0.1	2048	2.1e-4	12	2.3e-3
22	0.776	0.1	2048	2.8e-4	12	2.1e-4
23	—	0.12	2048	2.5e-4	12	2.7e-3
24	0.603	0.1	2048	2.6e-4	12	2.2e-4
25	0.799	0.13	2048	2.4e-4	12	2.7e-3
26	0.771	0.14	2048	5.2e-4	12	2.7e-3
27	0.621	0.21	2048	1.1e-4	12	6.7e-4
28	0.78	0.15	2048	2.2e-4	12	4.3e-3
29	0.752	0.13	2048	4.6e-4	12	8.8e-3

Table 4: Summary of the Optuna hyperparameter optimization for the *EnhancedPeriodClassifier*.